ARI TECHNICAL REPORT

19 TR-78-A10

Versus

6

Authoring Systems Vs Authoring Languages for Instructional Systems Development: Implications for Department of Defense

by

10

C. Victor Bunderson

BRIGHAM YOUNG UNIVERSITY

Contracted by:

BATTELLE COLUMBUS LABORATORIES

Columbus, Ohio

11 SEPTEMBER 1978

12 46p.

Contract DAJC04-72-A-0001

15

16 2Q763731A762

Beatrice J. Farr, Project Scientist
Leon H. Nawrocki, Work Unit Leader
Educational Technology and Training Simulation Technical Area, ARI

Prepared for

ari

U.S. ARMY RESEARCH INSTITUTE
for the BEHAVIORAL and SOCIAL SCIENCES
5001 Eisenhower Avenue
Alexandria, Virginia 22333

U. S. ARMY RESEARCH INSTITUTE FOR THE BEHAVIORAL AND SOCIAL SCIENCES

A Field Operating Agency under the Jurisdiction of the Deputy Chief of Staff for Personnel

Research accomplished
under contract to the Department of the Army

Battelle Columbus Laboratories

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER TR-78-A10	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* AUTHORING SYSTEMS VS AUTHORING LANGUAGES FOR INSTRUCTIONAL SYSTEMS DEVELOPMENT: IMPLICATIONS FOR DEPARTMENT OF DEFENSE		5. TYPE OF REPORT & PERIOD COVERED 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) C. Victor Bunderson		8. CONTRACT OR GRANT NUMBER(s) DAJC04-72-0001 (Task-Order-74-424)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Battelle Columbus Laboratories Columbus, Ohio		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 2Q763731A762
11. CONTROLLING OFFICE NAME AND ADDRESS Office of the Deputy Chief of Staff for Personnel Washington, DC 20310		12. REPORT DATE September 1978 13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)* US Army Research Institute for the Behavioral and Social Sciences 5001 Eisenhower Avenue, Alexandria, VA 22333		15. SECURITY CLASS. *(of this report)* Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

Research monitored technically by Leon H. Nawrocki and Beatrice J. Farr, Educational Technology & Training Simulation Technical Area, ARI.

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Computer-Based Instruction
Programming languages
Instructional Systems Development (ISD)
Authoring systems
CAI

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

This is one of a series of Technical Reports dealing with the authoring process and Computer-Based Instruction (CBI) and Artificial Intelligence. The distinction between authoring systems and authoring languages are noted, and the major goals for effective authoring are set forth. Critical goals include the need to (1) reduce the costs for authoring and production of computer-based instruction materials and (2) maintain or increase the quality of these materials. Primary constraints are the high turnover of authoring personnel and the need for authoring systems (for military purposes) to fit

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

20. (continued)

within the framework of the current Instructional System Development (ISD) model.

Accession For	
NTIS GRA&I	✓
DDC TAB	
Unannounced	
Justification	
By	
Distribution/	
Availability Codes	
Dist.	Avail and/or special
A	

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

FOREWORD

This work was performed as part of the Army Research Institute's (ARI) research program on the application of computer technology in education and training. The effort was initiated and funded during FY 75 within the Unit Training and Educational Technology Technical Area. under the direction of Dr. Frank J. Harris, Chief, and Dr. Joseph S. Ward, Work Unit Leader. Responsibility for completing and documenting this work was assumed by the Educational Technology and Training Simulation Technical Area during FY 76. Acknowledgement is extended to Dr. Beatrice J. Farr, the conference coordinator, who also served as editor for all technical papers, and to Dr. Leon H. Nawrocki, who chaired the sessions.

The primary impetus for this undertaking was the nearly universal belief among members of the DoD research community that there was a need for more interaction between those engaged in research, those involved in creating software and developing authoring languages, and hardware vendors. One unfortunate consequence of this lack of communication was that system requirements for users (authors) were frequently overlooked to the extent of being detrimental to system effectiveness. In an attempt to ameliorate this situation, ARI convened a three day meeting so that selected representatives from each of the above mentioned domains could discuss developments as well as problems of mutual interest. The conference had multiple goals; it was directed toward facilitating information exchange and toward establishing suitable guidelines for applying computer technology to training needs, with military training as the focal point.

Through the Scientific Services Program of the US Army Research Office, a contract was let under Battelle Columbus Laboratories to secure the services of ten scientists and educators currently engaged in widely diversified CAI activities. These experts, as well as technical and user representatives from each of the services psychological research organizations or operational CAI activities were the primary conference participants. In addition, more than fifty individuals from the Department of Defense, other government agencies, private research groups and academia were invited to attend the first day of the meeting as observers. The conference was held 9-11 September in Alexandria, VA. During the first morning session representatives from the Army, Navy, and Air Force gave formal presentations detailing both past and present programs relating to computer-based training. Considerable attention was also focused on current and anticipated problem areas. The afternoon was devoted to information exchanges between the participants and observers. The remaining two days were spent in small-group problem-solving sessions which culminated in decisions regarding the topics of papers to be prepared by the participants subsequent to the meeting.

As initially envisioned, the working sessions were expected to focus almost exclusively on the authoring process. Although the major emphasis did remain as planned, during the course of the conference it became clear that it would be more profitable to expand the scope beyond the original conception. In effect, the new agenda encompassed topics ranging from models which describe students, instructors and the learning process to sophisticated problems in artificial intelligence.

One of the primary goals of the conference was to attempt to have this diversified group of experts arrive at some consensus with respect to: defining user needs and requirements for authoring languages, identifying deficiencies within existing languages, delineating desirable characteristics for an ideal authoring language and establishing priorities for future research. Although somewhat less than consensus was reached, participants did identify a number of the most critical issues and offered guidelines for research directed toward resolving the major problem areas.

Joseph Zeidner

JOSEPH ZEIDNER
Technical Director

AUTHORING SYSTEMS VS AUTHORING LANGUAGES FOR INSTRUCTIONAL SYSTEMS DEVELOPMENT: INPLICATIONS FOR DEPARTMENT OF DEFENSE

BRIEF

Requirement:

This paper is the third in a series of reports emerging from a conference on Authoring Languages and Research Problems in Computer-Based Instructional Systems. The conference was sponsored and conducted by the Army Research Institute for the Behavioral and Social Sciences as part of the Technology Base Work Program. The effort was included in the "DoD Integrated Plan for the Use of Computers in Education and Training".

Approach:

A three day conference was convened to determine the state-of-the-art and future research directions for authoring systems, particularly research issues relevant to improving the interface between computer based instructional systems and instructional developers (authors). Participants consisted of ten technical consultants who were charged with identifying and reporting on major topic areas. Additional invited technical and user representatives (governmental, industrial and academic) participated either actively or as observers throughout the conference. A list of participants is provided at the Appendix. The first day was devoted to (1) formal presentations, by military training system representatives, describing current and planned computer-based instruction activities within the military, and (2) a roundtable discussion which delineated and defined major topic areas to be addressed. During the following two days, participants divided into four working groups. At the final session, each group presented a summary of their discussions relating to the key issues and approaches to authoring system research. Active participants selected topics for follow-on reports to be prepared subsequent to the conference.

Determinations:

Authoring systems differ from authoring languages in that the latter deal primarily with the mechanical boundary conditions that permit packaged courseware to run on a particular computer. Such conditions include the language requirements for: (1) display creation, (2) response acceptance, (3) analysis of constructed response and, (4) conditional branching. Authoring systems deal with broader concerns, namely the critical concepts and variables involved in the process of courseware development as a whole - including curriculum design, authoring, production and revision. From the requirements articulated by user organizations, two major goals for acceptable authoring systems have been inferred, along with a related set of constraints. The critical goals are the need to (1) reduce the costs for authoring and production of computer-based instruction materials and (2) maintain or increase the quality of these materials. Primary constraints are the high turnover of authoring personnel and the need for authoring systems (for military purposes) to fit within the framework of the current Instructional System Development (ISD) model.

Utilization of Findings:

The goals and constraints detailed in this report are of use to those concerned with the specification of computer-based instruction system requirements. Moreover, the proposed system criteria should serve as the basis for the development of author aids to facilitate the application of the ISD model.

CONTENTS

LIST OF TABLES

AUTHORING SYSTEMS VS AUTHORING LANGUAGES FOR INSTRUCTIONAL SYSTEMS DEVELOPMENT: IMPLICATIONS FOR DEPARTMENT OF DEFENSE

INTRODUCTION

In this section, the opportunity for improving training in the Department of Defense and reducing its costs is discussed, and the problems in authoring which block this opportunity are described. At a conference on Authoring Languages sponsored by the US Army Research Institute, informal presentations by Don Kimberlin from the Computerized Training System, Ft Monmouth, NJ, George Leahy and Dexter Fletcher from the Navy Personnel Research and Development Center in San Diego, and Ed Gardner, from the Air Force Human Resources Laboratory, Lowry AFB, Colorado, outlined the state-of-the-art as well as the research gaps existing in the services at present.

The second section of this paper outlines the basic thesis. Namely, that it is important to make a distinction between an authoring system and an authoring language. Most authoring languages, as they have evolved over the past twelve to fifteen years, do not include the semantics required by authors involved in the systematic instructional design process. Since every service utilizes a common tri-service model of a systems engineering approach to instructional development, it seems appropriate that authoring systems, including their computer-related components, should facilitate the performance of the tasks prescribed by this model.

In the last section of this paper, the current and potential contributions of computer-aid in design are discussed. Reference is made to computer-aided design in engineering and architecture.

Department of Defense Needs for Authoring Support

Computer-assisted and computer-managed instruction offers opportunities for cost-competitive benegits in all of the services. Research conducted in the Department of Defense and in the civilian sector has overwhelmingly shown equal or greater performance gains by CAI students and increased throughput. Some data collected by the Army (Kimberlin, 1974) showed improved throughput, equal achievement, and better retention and transfer of learning. In addition, 21% less failure, 35% less time, and more favorable attitude on the part of trainees using CAI was reported in the same study. The Navy has reported as much as 39% - 54% time savings through the use of CAI (Lahey,1974). The Air Force's AIS project, according to Gardner (1974), had set a goal of 25% improved throughput, but existing off-line individualized versions of the course ultimately to be mediated by CAI and CMI have already shown 35% improved throughput. Increased throughput can be translated into cost savings in student's salary and living expenses (given more flexible duty-assignment procedures for training billets). In addition, cost per student goes down for a given investment in physical plant and training personnel.

Data on CAI effectiveness shows greater success, increased motivation, and a more positive attitude for lower ability students. This has considerable relevance for all-volunteer services.

There is a strong need to capitalize on the cost-advantages of electronics technology. The dramatic decline in the cost of electronics components offers new alternatives for substituting CAI for more expensive uses of equipment and personnel. The Department of Defense, in particular, can promote cost-effectiveness by using CAI to reduce the requirement for time spent on expensive equipment simulators and time spent in training missions using expensive ordance such as aircraft, tanks, and other devices.

There are a number of problems and constraints which have prevented the full utilization of computer-assisted instruction. The cost and effectiveness of CAI authoring is one of the major problems, and is the issue which prompted the conference from which this paper emerged.

In his review of Army needs, Kimberlin stressed the problem of staff turnover. Army authors typically have more than twelve years' education, and average rank of E-6, and twelve to fifteen years of experience. They are usually over thirty years of age. Their duty assignments, however, are constantly changed and there is no guarantee that an author trained to proficiency will remain long with a CAI development project. The civilian authoring personnel, who may have longer tenure on an instructional development project, may have had no more systematic training in authoring than that provided by miscellaneous workshops. Their average education is fourteen years, they have had an average of twenty years of experience and are generally over forty years of age.

Army authors typically must produce the courseware essentially by themselves (except for some internal military support). They cannot alter the content substantially, and they must stress production first, with creativity and the "gee whiz" aspects of CAI taking a back seat. To accomplish this they frequently exploit canned macros to facilitate production efficiency. It is my belief that the author should be his own coder in the Army, although bulk entry can be done by clerks.

Lahey reported that Navy authors, who have a three year tour, are expected, within eighty hours, to be able to produce a linear program (including text and graphics). Within 160 hours they should be able to produce a program with sophisticated branching. This rate of production is expected to occur after what is, at most, three months of training. Because of training time and turnover, full-utilization of authors for more than one to one and a half years is unlikely. Leahy reported that it still remains an issue in the Navy whether to use the author as a coder or to differentiate the staff to include authors and separate coders.

Gardner described the AIS Project and the CAMIL language as a response to Air Force authoring needs. He did not speak to the issues of training requirements for Air Force CAI authors nor to the problem of turnover, but did emphasize the need for transportability. He stressed that the definition of CAMIL includes design for machine independence. The problem of transportability was addressed by the other DoD personnel as well. Kimberlin, for example, stated that although transportability was critical, the Army would probably be satisfied if courseware could simply be transported to like systems.

Kimberlin provided details on the amount of training time it is expected a novice will need before beginning to use an authoring language. He stated that, using existing macros, an author must be able, at the end of two weeks, to create the text for a lesson. Author training in entry and editing should produce proficiency in most of the conventions within two days and in all conventions within two weeks. At the end of three months, he must be able to develop any kind of lesson, test, or off-line mix. He will also be able to design his own new sub-routines or macros. At the end of six months he would be fully proficient and would be able to design a lesson which doesn't follow an existing model. Kimberlin stressed that the author would be likely to be transferred at the end of two years or less.

Design Requirements for CAI/CMI Authoring Support

From the requirements noted previously, together with information garnered from published Department of Defense documents, it is possible to infer two major categories of goals, and a related set of constraints, with regard to the type of authoring systems that would be acceptable to DoD in achieving these goals. These goals and constraints are listed in Table 1. They form the criteria toward which proposed new authoring systems may be designed, and against which existing authoring languages and systems may be evaluated.

The categories in Table 1 are compatible with the requirements given by each of the DoD speakers, but each service may place different emphasis on each goal and constraint category. The Army and Navy have already done this in some sense by describing the requirements for training (subgoal 1.1) in terms of maximum man-hours and man-weeks required. Although subgoals 1.2-1.4 were not specifically mentioned, these categories of cost reduction were implicit in the presentations.

TABLE 1

Design Requirements for DoD-Oriented Authoring Systems

GOAL 1. To reduce the costs for authoring and production of CAI, CMI, and related materials of instruction (MOI).

- 1.1 To reduce training time required before an author can produce usable MOI.
- 1.2 To reduce production time for standard components (lessons, scripts, etc.) for trained authors
- 1.3 To reduce the costs of revision and updating of MOI.
- 1.4 To reduce the costs of maintenance, duplication, etc., of in-use materials.

GOAL 2. To improve the quality of authored MOI.

- 2.1 To reduce the costs of training in DoD job categories in comparison to existing alternatives.
  - 2.1.1 To reduce failure rate.
  - 2.1.2 To increase throughput.
- 2.2 To maintain parity or increase measures of instruction effectiveness of MOI in comparison with existing alternative.
  - 2.2.1 Using end-of-unit and end-of-course measures.
  - 2.2.2 Using standardized retention measures.
  - 2.2.3 Using standardized transfer measures.
  - 2.2.4 Using on-the-job performance measures.
  - 2.2.5 Using measures of attitude and affect.

GOAL 3. Constraints on designs for achieving Goals 1 and 2.

- 3.1 There is a high turnover in DoD authoring personnel.
- 3.2 The educational level and taste level of DoD authors requires a pragmatic, common-sense approach. Academic theory and jargon are seldom helpful.
- 3.3 An authoring system must fit within the framework of existing DoD Instructional Systems Development (ISD) model.
- 3.4 For CAI/CMI materials, the design should minimize the cost of transporting courseware from one computer configuration to another.

All three services spoke to the issue of improving quality (as measured by increased throughput) and some mentioned reduced failure rate, as well as various other of the subgoals listed under 2.2.

The constraint of turnover (3.1) was mentioned by all. Parameters for the population of authors (constraint 3.2) were specified by both the Army and Navy.

The constraints demanded by existing ISD models may be found in the regulations which guide each DoD training development organization, particularly the ISD model developed by Florida State University under the direction of the Interservice Training Review Organization (called the "ITRO" model).

The constraints of transportability (3.4) is a function of the state-of-the-art in CAI/CMI hardware and software. There are different systems, not just between, but also within the services, and each CAI or CMI system may have different versions at individual installations. The prospects for ready transportability will have to await standardization in text, media, and computer components of courseware, as well as in software and hardware. Since such standardization does not appear imminent, the more modest requirement (3.4) of minimizing the cost of transportation is the only feasible constraint. It is a realistic constraint, since different authoring systems can vary widely in the cost of transporting courseware from one system to another.

A Distinction Between Authoring Languages and Authoring Systems

The term "authoring language" has as its referents a set of so-called

CAI languages, (some of which are general purpose languages). These have been catalogued and listed by Zinn (1969), Frye (1969) and others. The semantics of these languages deal, not with the critical concepts and variables involved in the process of courseware development as a whole, but primarily with the mechanical boundary conditions for packaging courseware so it will run on a particular computer.

By "mechanical boundary conditions" is meant the language requirements for (1) display creation, (2) response acceptance, (3) analysis of constructed responses, and (4) conditional branching. These are a reflection, in the first two instances, of the capabilities of the terminal device to present different types and formats of displays and to accept different types and formats of inputs. In the last two instances the "boundary conditions" reflect the available algorithms and data structures for the recording and analysis of student inputs, and sequencing divisions based upon these inputs and authored program structures. A fifth "boundary condition", implicit in the earlier "authoring languages", is the data structures and algorithms available for the storage and retrieval of a library of content data.

While the term "authoring language" may be rehabilitated in the future to refer to more of the semantics of courseware design, authoring, production and revision, it is more meaningful, at this time, to use the term "authoring system" to refer to these broader concerns. Authoring language as defined above, then, becomes a subset of an authoring system - the subset dealing with the packaging, entry, and debugging of the on-line components of a systematically designed set of instructional materials.

An authoring system, on the other hand, would include off-line as well as on-line aids to the author. In particular, it would include a system for initial training of new authors, or system for on-the-job training and prompting of authors, formatted manuscripts for facilitating the writing revision and later moderation of various instructional components, and a management system for tracking progress in the various stages of design, authoring, production, and revision. Many of the steps now performed off-line may eventually be facilitated by computer aids to design, authoring, production, and revision. These computer aids, however, go well beyond the capabilities of past and present authoring languages. They will involve the semantics of systematic approches to courseware development as a whole, not now found in existing languages, and the semantics of analytical approaches to instructional design, only recently emerging.

Mechanical Boundary Conditions: The Semantics of Existing Languages

The term "semantics" has been used above to highlight the distinction between possible meanings which can be communicated readily by the use of the primitive concepts and terms of a particular formal language or system. Any language, natural or formal, has a set of elements (words or basic symbols) which, according to a set of grammatical or syntactical roles, can be combined into a set of formally correct sentences. Meanings can be conveyed by these sentences, some readily, with ease and precision, others with difficulty and obtuseness, and still others not at all. The meanings which can readily be conveyed consitute the semantics of the language. The argument that particular activities can be performed in a given language, albeit with obtuseness, is not a defense against the criticism that it lacks the proper semantics.

Language gives a person power over his environment. When children gain language ability they become their own paymasters freeing themselves from environmental control, while gaining in personal control. As a field of knowledge becomes well specified, a formal language can be developed and expert practitioners can gain more control and improve their efficiency through use of that language. The field of simulation, for example, is well characterized by the primitive constructs within the language SIMSCRIPT. After learning the internal and external event variables and their relation to time flow, a programmer in SIMSCRIPT can talk simulation and think simulation. A programmer in APL can think applied mathematics with emphasis on array manipulation in a manner which far exceeds the efficiency with which he could think prior to Iverson's (1962) synthesis of the notation across several fields of applied mathemathics. But what of instruction? Are QU's, CA's, WA's and UN's[1], the primitives of a language of instruction? Can authors quickly be trained to think in terms of instructional variables that matter when the language they use does not have the proper semantics?

That the answers to these questions is <u>no</u> is well testified to by a decade of experience in a wide variety of CAI projects. The reasons are

[1]IBM's Coursewriter language introduced these commands for QUestions, Correct Answers, Wrong Answers, and UNexpected answers. These primitives are found in one form or another in all so-called authoring languages.

explained in more detail in the next section of this paper. Table 2 lists the mechanical and processing requirements which form the basis of most CAI languages. The entries in Table 2 are not intended to be exhaustive, merely illustrative. The five categories displayed are not intended to be "fair" to existing languages, since most have added new features not found in Table 2, most notably data-recording and report-generating routines for the formative evaluation of CAI lessons, and macro definition facilities (a macro can be designed around good instructional variables and can incorporate validated instructional paradigms). A set of macros can provide a basic vocabulary of instructional strategies, if properly constructed. Finally, the text and graphics editing systems (of varying degrees of sophistication) that have evolved around some existing languages, impinge importantly on the tastes of authors.

Rather than being "fair" to existing languages, Table 2 is designed to show the concerns which were first addressed by the authors of many languages, for these concerns led to the creation of the data structures and algorithms which limit what can be done conveniently. To the extent that the initial design is limited in its ability to express the proper meanings, later additions will be limited and require obtuse thinking.

Table 2 shows the semantics content of most CAI languages. They deal (in categories 1 and 2) with the mechanics of communication through a terminal device having particular display and response mechanisms. Great elaboration can occur within any of these categories. For example, the semantics for graphic display are evolving nicely as the authors of languages develop vocabulary and associated data structures for creating

TABLE 2

Mechanical Boundary Conditions:
The Semantics of Most Existing CAI Languages

1. Ability to control the display of content on any of the following which are mechanically pcssible given the display hardware:

   - Typewriter or teletype
     - Up and down 1/2 index
     - Interchangeable font
   - Time displays
   - Image projector
   - Audio
     - Play or record
     - Access random addresses
     - Compile utterances from digital code
   - CRT or Plasma Display
     - Character display
     - Digital-stored graphics
     - Vector-generated graphics
     - Hard copy from display
     - Videotape or videodisc display

2. Accept and compare responses which are mechanically possible:

   - Keyboard
   - Response Latency
   - CRT pointer
   - Touch-sensitive surface (over-image projector or CRT)
   - Other (drawing, voice, kinetic)

3. Process contructed responses:

   - Match transformed response to target
     - Edit-insert or delete any characters
     - Keyword or keyletter scan
     - Phonetic
     - % character match
     - Numeric tolerance
     - Algebraic equivalence
     - Transform and match to "deep structure" representation
   - Synthesize feedback message
     - Process in relation to a target
     - Synthesize message from processed response
     - Display contents of buffers and counters

4. Execute conditional sequences:

   - Label lines or blocks of code
   - Record student response history
   - Computer logical or mathematical conditions from response records
   - Provide built-in or programmable logics
   - Process lists of labels

5. (Implicit in 1., above) Ability to store and retrive content for display

and manipulating graphics. In the area of analysis of constructed responses, computer scientists have developed sophisticated algorithms for interpreting subsets of natural language and mathematical expressions entered by students. These are important areas for general uses of computers in man-machine communication, but still do not involve much of the semantics of prescriptive instructional design. The contribution of todays CAI languages in the display and response analysis areas is analogous to providing a classroom teacher with a rapid and flexible graphics artist and an expert "listener" to better understand the student's questions. It still does not impinge on the content and sequence of instructional displays.

Semantics for instructional strategy on sequence control are most notably lacking in existing CAI languages. While it is true that flexible conditional branching commands (category 4 in Table 2) permit the implementation of any strategy, the possible strategies are virtually infinite, so that most successful authors must, of necessity, go to standard patterns. Unfortunately, the most common standard pattern, the "tutorial", defined by the intrinsic branching of Coursewriter and copied in numerous languages, is atheoretic and low yield in terms both of authoring efficiency and good results with students. The standard tutorial sequence consists of asking questions, anticipating answers and providing feedback. It is a synthesis of Norman Crowder's intrinsic programming (where branching was done on multiple choice responses) and Skinner's linear programming (where constructed responses were possible). Tutorial CAI consists of an intrinsic program with constructed response. It is modeled after a format, not after an instructional rationale. It does not consider the class of learning, whether memorization, concept learning, rule-using or problem-solving, for which the conditions of learning differ. The examples usually given in authoring guides are at the memorization level, and even there they are inappropriate tactics for teaching memorization behavior.

It is depressing to see the poor examples which are presented in author training manuals. So many of them are the sequences which emerge from tutorial CAI; and while it is agreed that an author can use these languages to create the standard tutorial strategy, the point is that the patterns available to the author as examples are built in by implicit branching and provide him with a semantics of instruction which he will probably use without giving it much thought.

The last category in Table 2, "Ability to store and retrieve content for display," is a dimension of authoring languages which has been largely neglected, and is probably the single most fundamental concern in the design of authoring aids. It deals with the structure of the content data base. Roy Kaplow's paper (in press) illustrates several extremely useful aids to authoring, debugging, and revision, which emerge from the creation of a structured data base, accessible in multiple ways. The systems he and his colleagues have designed represent a substantial advance in incorporating semantics that are more central to the authoring processes than those found in most existing authoring languages.

Item 5 in Table 2 states that the ability to store and retrieve content for display is implicit in category 1, the ability to control the display of content in any of a variety of devices. Authoring languages like Coursewriter, PLANIT, TUTOR, PILOT, etc., build content into the display commands themselves, so that content and sequence strategy are inseparably intertwined in a long listing of code, with ad-hoc labels. Because of this, getting back to any content component, revising any sequence strategy, cross-referencing, counting like components, and providing management data are all very difficult processes.

Results of Inadequate Authoring Semantics in Existing Languages

Using extensions of existing languages may offer no ready route to achieving the goals specified earlier in this paper because these languages are themselves a part of the problem. Their deficiences exist at two levels: content and program structure, and instructional strategy. Most existing authoring languages provide no built-in structures, nor (as Kaplow's TICS system does) commands for creating, manipulating, observing, and documenting such structures. The results of this deficiency are unfortunate. A vocabulary of different higher-order structures for instructional systems is not likely to develop and computer aids for producing particular high-order structures do not evolve. Classes of perfect macros or subroutines which fit as modules within overall structures are slower to evolve and become useful (although the creation of macros or standard subroutines in existing languages has been the major development toward meeting the DoD needs listed previously). Revision of programs and management of production are greatly facilitated by the existence of structured content files that are coherently organized.

At the level of individual instructional strategies, the lack of proper semantics creates additional difficulties. Frequently, authors fail to learn a vocabulary and syntax for expressing well-formed instructional propositions having instructional validity. What happens, all too often, is that they fall into a pattern of emulating sequences of doubtful instructional utility and questionable overall program structure. They spend the bulk of their initial training time learning coding and programming. It then requires continual creativity and skill to program the kinds of CAI sequences which have become favored in existing CAI lanaguages. This creativity and skill may well be independent of the creativity and skill required for authoring materials which teach efficiently and effectively. This creativity is too often squandered on clever display algorithms and clever response analysis and feedback schemes.

Weaknesses in the <u>tutorial</u> model have already been discussed. Simulations and games, also popular in many circles, have instructional weaknesses as well. They function best when the basic concepts and principles are understood by a student, but authors of simulations frequently neglect to teach them. Students who are above average in intelligence can discover these underlying concepts and principles inductively, and may already be endowed with the verbal information required to disambiguate the instructions and contents of games and simulations. However, lower ability students often find this discovery mode of learning to be less efficient and effective (Bunderson, 1971). If they are given instructional approaches which incorporate the proper variables, these lower ability students can also achieve and can gain the motivation and enjoyment good CAI always brings. Positive steps toward dealing with this problem have been taken at many installations. Programmers have been hired to work side by side with authors. Preprogrammed paradigms, of varying instructional efficacy, have been provided to give the authors some of the instructional vocabulary they need.

ISD Models and Instructional Components: The Semantics of Needed Authoring Systems

One approach to meeting the requirements specified in the section of design requirements is to create, for authors, a formal vocabulary of instructional development, implemented by a set of concepts taught in a training program, and incorporated in formatted manuscripts, management aids and procedures, and computer aids. This vocabulary will enable authors to think and talk about the important activities and products of an accepted Instructional Systems Development (ISD) model, and about the instructionally relevant components of each intermediate and first product. The current DoD ISD model was referenced previously. Earlier versions of this model may have varied one from another in terminology and in the emphasis given to any particular stage, but the same three basic stages were found in all iterations. The process involves (a) some form of problem analysis that focuses on desirable real-world performance and results in specific statements of instructional objectives, the achievement of which is measurable; (b) a design and development process that insures instructional materials (displays, frames) directed at fostering achievement of the stated objectives, using the most efficient and effective instructional strategies or techniques; and (c) an evaluation-validation

TABLE 3

Examples of Substeps Under Three Phases
Common to all DoD Instructional Systems Development Models

A. Problem Analysis Phase

1. Perform overall problem analysis
2. Conduct job analysis survey
3. Select tasks for training
4. Perform task analysis and produce objectives and hierarchies

B. Design and Development Phase

1. Sequence and group objectives, providing a coherent tabling scheme
2. Make media decisions for each group of objectives
3. Develop lesson specifications
4. Develop manuscripts, storyboards, etc., for each lesson and component
    * Workbook
    * Slides/tape
    * Trainer exercise
    * Lab experiment
    * CAI lesson
    * Videotape or movie
    * Lecture lesson plan
5. Provide instructional and content review
6. Package manuscripts, storyboards, etc., in prototype and final form

C. Evaluation/Validation Phase

1. Prepare formative and summative evaluation plans
2. Throughput the design and development phase, collect formative data and revise
3. Implement Instructional System
4. Conduct validation study
5. Complete scheduled revisions

phase. During this third step, students who are representative of the intended target population use the program and then are evaluated with respect to their achievement of the stated objectives; changes are made in the program wherever the students do not achieve the objectives, and the evaluation-revision process is continued until the program demonstrates that it does indeed have a measurable and consistent effect on the behavior of students who meet entry-level requirements.

Table 3 summarizes some of the steps which occur under each of these three major headings. Not all ISD models for the Military employed all of the substeps listed in Table 3. In particular, steps B2, B3, and B4 were often lacking in detail in a given ISD model (B3 was often lacking completely). This is the heart of the authoring process, and is presently an area where art and tradition are more involved than systems engineering.

Three approaches to design and development are used in military training (as well as in civilian ISD). These may be characterized as the artistic, the empirical, and the analytical approaches. Bunderson and Faust (1976) have described these briefly. The artistic model is the approach generally favored by persons from the publishing, still and motion photography, and educational and broadcast television communities. The tradition for style and taste which have developed within these communities sets the standards. Products with positive reviews - or awards - become the models to emulate. In CAI, this "medium is the message" approach of the true artist has strong proponents, especially on systems with attrative graphics capabilities and powerful language features, such as the PLATO system.

PLATO is also one of the more attractive systems to proponents of the empirical approach, because the TUTOR language has some useful features for collecting student data and providing rapid feedback to authors. It is not surprising that CAI languages which lack underlying instructional theory have developed tools to support the atheoretical empirical approach. The empirical model gives no guidelines for how to produce the first draft of a lesson, but suggests that the first crude efforts be taken to a few students, and revised on the basis of student reactions as often as necessary. The revisions may be founded on intuition, or mere caprice, but they must continue until the lesson achieves its objectives.

The analytical approach is the most recent to be applied in military training. It is based on prescriptive instructional principles that start with a classification of objectives. Recall objectives, classification, and rule-using objectives fall, in general, into a hierarchy, with recall at the bottom and role-using at the top for closely related content. This regularity may be used in producing lesson hierarchies and in checking for the completeness of such hierarchies. Instructional components called "maps" or "outlines" may be produced as a result of the task analysis process which yields these objectives. Recall, classification, and rule-using objectives also differ in the instructional variables which are most effective for each. These variables can in turn be implemented as instructional components (e.g., rules, examples, practice problems, practice feedback, and helps). Each of these components has formal properties whose presence or absence makes a great deal of difference in student learning. Unlike the empirical approach, the analytic approach gives prescriptive guidance to the author for how to write the initial draft of any component and how to revise it if evaluation shows it to be deficient.

The analytic approach thus provides prescriptive guidance and two levels of generality: one guiding the development of content structures and sequences (steps B4 and B1 in Table 3), and the other guiding the selection or design of instructional strategies and tactics (steps B2 and B3).

The authoring system implemented in TICCIT[2] is the first application of this analytic approach to CAI. The basis of TICCIT's design rests in principles of instructional psychology and cybernetics, described in the chapter by Bunderson and Faust in the recent NSSE Yearbook on the Psychology of Teaching Methods (Gage, 1976). The work of Merrill and his co-workers formed the propositional basis for the componentized data structure upon which the TICCIT authoring system is built. Merrill and Boutwell (1973) produced the foundation paper which set forth the taxonomy of instructional variables that led to the design of TICCIT's MAP, RULE, EXAMPLE, PRACTICE, and HELP components. More recent papers are by Merrill, Olson, and Coldway (1976), and Merrill and Wood (1975).

The componentized approach to courseware development first implemented on the TICCIT system for about 600 objectives in freshman mathematics

---

[2]TICCIT stand for Time-shared, Interactive, Computer-Controlled Information Television. The hardware and systems software were developed by the MITRE Corporation. The authoring system, much of the author and user software and courseware in mathematics and English was developed at Brigham Young University, with National Science Foundation funding.

and English has been extended into military training for upwards of 30,000 objectives in all of the media forms listed under B4 in Table 3.

Projects where this approach was used included flight training for the Navy's S-3A aircraft, the SH2 Helicopter, the Navy's P-3C aircraft, the Marine Corp's Mobile Training Teams, and a variety of other jobs. The two TICCIT systems now installed at North Island, San Diego, and Cecil Field, Florida, have already exceeded, by a large margin, the number of objectives of CAI instruction implemented on TICCIT for college courses. These 30,000 objectives have been produced by military authors under all of the constraints listed under 3.0 of Table 1, including rapid author turnover.

Transportability has been demonstrated from the componentized manuscripts to other media, but not yet to other CAI systems. The coherent data structures which separate content files from branching strategies provide a promising basis for minimizing the cost of transporting from one system to another. Content files (e.g., MAPS, OBJECTIVES, RULES, EXAMPLES, PRACTICE, FEEDBACK, and HELPS) could be translated automatically to compatible display devices, and the branching logics for maps, tests, primary instruction, advisor, and answer-processing, could be reprogrammed on the new host computer just <u>once</u>, rather than having to be redone with each new section of similar code.

The TICCIT authoring system, based on the analytic model, may be constrasted to other approaches for producing course structures and within-node strategies suggested elsewhere. Kaplow's TICS system appears to be the most general. It appears that the TICCIT MAP structures could readily be implemented within TICS and its descendants, along with other forms of task structures. It is not the MAP logics on TICCIT however, but the vocabulary and syntax inherent in the analytic model that makes it easy to train authors to produce TICCIT MAPS.

Peters (1974) has suggested that systems like the instructional dialog facility developed at Hewlitt-Packard can draw the information out of an author and assist him to formulate it in representational forms which lend themselves to the task of lesson development. More ambitious is Stelzer's (1974) suggestion that machine intelligence be used as an "intelligent partner" or "expert consultant" to interact with the author in the formulation of a content structure. As Stelzer points out, this interaction often results in discoveries about the nature of the subject matter. It would be a major contribution if this optimism were vindicated and computer systems began to help authors perform this high level task in the not-too-distant future.

While some scepticism must be held about the near term availability of aids such as those suggested by Stelzer and Peters, it is clear from experience on the TICCIT project that the computer can assist in the training, management and production of objectives, heirarchies, and other forms of task structures. The key is the extent to which the computer primitives convey the semantics of instructional science. The TICCIT MAP logic presents, to the student, a learning hierarchy display with the prerequisites on the bottom and a test at the top. In between are the instructional elements which lead from the prerequistes to the capability measured by the test. The MAP logic is applicable at three different levels of decomposition. The COURSE MAP has, as its elements, instructional UNITS. A UNIT MAP has, as its elements, LESSONS, and a LESSON has SEGMENTS. A segment teaches one objective, usually a single concept, rule, or recall objective. The MAP logic gives the student the ability to survey, for it is an interactive display which allows him to look at introductory materials consisting of videotapes, or linear overview sequences using graphics and audio. He can also look at the objective of each element of the MAP, which may be a unit objective, a lesson objective, or a segment objective. The MAP logic also provides the student with status information by coloring the boxes red, yellow, or green depending upon how he has performed in the instructional material which underlies a MAP box.

The MAP logic is a great aid in training authors and assisting them in production, for it makes explicit the decomposition process of a course structure. Once MAPS have been created, it is clear to everyone what the production task is. The MAP structures themselves can be used as production management tools, forming an organizing structure for scheduling the various steps of production on each instruction component and checking off the completion of each scheduled production activity.

MAP logics and the associated TEST logics that go with them also facilitate a production procedure which has been called "lean development". For CAI it means that the first stages of production can result in a set of MAPS, RULES, TESTS, and overviewing which, for the brighter students, are sufficient for them to complete the entire course. Data can be collected on students who go through this very lean CMI-oriented version (any off-line printed materials which may exist can be used for early tryout without additional production). This data can lead to an understanding of which lessons and objectives are most and least important, and which will require the greatest emphasis in the instructional production process.

Some Known Computer Aids to Authoring

This section will review some requirements for that part of the ISD process dealing with the specification of content structures and lesson strategies, the area wherein the analytic model promises to make its greatest contribution.

Fundemental to the design of truly helpful and intelligent authoring systems is the requirement for a coherent, structured (componentized) content data base. This data base should identify the level of the content component (e.g. from TICCIT' course, unit, lesson, and segment) and the type of component (e.g. from TICCIT: MAP, RULE, etc.). A modular, hierarchical data structure (described in terms of instructionally meaningful constructs) aids in the process of training, and provides an effective means of utilizing short-time authors. It also contributes to the transportability (via the automatic translation of content files), and the entry, editing, debugging, and revision of content files. Additional data bases are required for algorithms which work upon the content files. This separation of content and strategy lends itself to cross-referencing, automatic file maintenance, and debugging tools, some of which are described by Kaplow (in press).

The contribution of current authoring languages can also be evaluated by reference to Table 3. The most telling criticism of these languages is the absence of structured content data bases. The greatest advantage comes in their facility for enabling step B6, the packaging of CAI manuscripts. Because of their on-line editing and display capability, in the future they can also impact on step B4, the actual writing of manuscripts. Some authors find it more effective to go directly to on-line materials rather than going through a manuscript phase. Revision and update are greatly enhanced, and formative evaluation data can be achieved more quickly. Support for evaluation, steps C2 and C4 are also strengths of some existing author languages, notably TUTOR.

Implicit in B3 is the final advantage of existing languages which is worth noting. Lesson specifications are based on pre-specified instructional paradigms and their components. Existing author languages offer excellent vehicles for the initial development and check-out of new instructional paradigms.

Computer Aided Design for Instructional Systems Developers

A true authoring system will use the computer at as many stages of the design process as it can effectively be used. In this section, some work in computer-aided design in fields other than instruction will be reviewed, and implications drawn for possible computer aids to the total instructional design process. Since DoD is involved in instructional systems development and not just CAI packages, it is appropriate that an authoring system assist individuals in the design and production of all kinds of instructional materials, and all of the intermediate products listed or implied by Table 3. It is interesting that the overall structure of the design process incorporated in the DoD's systems engineering models is analogous to the problem solving/design process in any of a variety of fields. For example, Gero (1973) reviews the stages of the design process used in architecture. It is similar in all significant points with the systems engineering model used in the DoD. Powers (1973) describes an analogous process used in chemical engineering for the design and production of physical systems. The analogs in design engineering in a variety of disciplines are numerous. Each of these authors are involved in Computer Aided Design (CAD), and use the computer at different stages of the design process.

The opportunity for substantial gains in cost-effectiveness through the use of the computer is available in all disciplines where a final product must be produced and used in a human environment. The investment in design is typically a smaller fraction of the total cost of production, yet it yields economic returns far in excess of the amount expended on it (Gott, 1973).

Powers (1973), points out, however, that there is a striking absense of computer aids in the definition of needs and objectives, and in relation to the synthesis of plausible systems. These areas are not well developed for either author languages or authoring systems as defined herein.

H. G. Adshead, manager of Design Automation in Manchester, England, stresses that design should be totally integrated with production (Adshead, 1973). He gives examples of design automation and computer aided design of digital systems, and shows how the computer can facilitate the development of design specifications as well as production output itself. This has relevance to the lesson-specification step (B3) of Table 3, and the production steps B4 and B6.

Computer aided design in engineering is based on mathematical models and programming techniques which permit graphic display of the consequences of alternate designs. With instructional design, we need to develop models of students and of the effects of strategy variables upon students who have difference characteristics. These models can permit us to predict student outcomes (time, performance, enjoyment, etc.) and so optimize designs initially with regard to student outcomes. We can also optimize the management of the total system by having models of the flow of students. queueing at the terminals, mixes of various media, etc. A simple optimization model of resource allocation at a gross level is provided by Atkinson (1972) who also has provided optimization models for instructional strategies.

These are only hints from this literature regarding where we should look for important progress in computer aided instructional design. The mathematical models in engineering permit the calculation of optimal solutions. Such models are primitive or non-existent in instuction, but the possible rate of simulators at both the macro level (course structures and media mix) and micro level (instructional strategy) are promising, and could lead, as Adshead stresses for circuit design, to a close link between computer aided lesson specification and component production. Some possibilities of this sort are implicit in recent advances in the analytic model discussed above.

A complete list of computer aids to the design of instruction would be very long. Including the simulators and production aids suggested by the CAD literature, such a list should include items found in Table 4.

It would be unthinkable for would-be designers of computer-based aids to authoring to go through the speculative exercise of forming a matrix with the descriptors in Table 3 as row headings and the descriptions in Table 4 as column headings. A few of the intersections of such a table are occupied by existing features of authoring languages. Such languages fit in the packaging, evaluation, and (weakly) the editing and revision steps. Other intersections are occupied by existing features of the TICCIT authoring software or the TICS system discussed by Kaplow.

Most of the intersections are empty, and many rightly so. Some computer aids suggested by the intersections of these two sets of descriptions, however, could be of substantial utility in achieving the goals specified in this paper's section on design requirements. Of particular promise, in that they would cut across almost all intermediate steps (at least wherever a document is produced as an intermediate product) would be entry and edit systems formatted around a set of validated conventions

TABLE 4

Varieties of Computer Support to
One or More Steps of the Design Process

1. On-line entry and edit systems for content files:
    - for text, either CAI or printed media
    - for graphics, either CAI or photographs
    - for audio
2. Data capturing systems for potential and actual users of instructional systems.
3. Data analysis and display systems.
4. Interactive entry and prompting systems for each intermediate design product of Table 3.
5. Simulators to predict the costs and payoffs of various media mixes, implementation strategies, and instructional strategies.
6. Design languages for task and content structures with documentation, cross-referencing, and debugging aids.
7. Design languages for instructional paradigms with documentation, cross-referencing, and debugging.
8. Management systems to provide scheduling and resource allocation control over the achievement of milestones within a defined content and component structure.

and components for the completion of each step. Associated author-prompting systems would facilitate on-the-job training and quality control. Along with these tools a management system, based on milestones for grossly described products for steps A1 - A3 of Table 3, and after that on milestones based on the content and component structure itself, could be the major tool for achieving the goals of reduced time and cost for initial authoring, under the constraints which exist within DoD.

APPENDIX

PARTICIPANTS

Mr. Avron Barr
Institute for Mathematical Studies in the Social Sciences
Stanford University, Ventura Hall
Palo Alto, CA 94305

Dr. Alfred Bork
Department of Physics
University of California
Irvine, CA 92664

Dr. John Brackett
SofTech
460 Totten Pond Road
Waltham, MA 02154

Dr. Victor C. Bunderson
Institute for Computer Uses in Education
Brigham Young University
Provo, Utah 84601

Mr. Frank Dare
CAI Project
USA Ordnance School and Center
Aberdeen, MD 21005

Mr. Wallace Feurzeig
Bolt, Beranek and Newman
50 Mouton Street
Cambridge, MA 02138

Dr. Dexter Fletcher
Navy Personnel Research & Development Center
San Diego, CA 92152

Dr. Ed Gardner
Air Force Human Resources Laboratory
Lowry AFB, CO 80230

Dr. Roy Kaplow
Massachusetts Institute of Technology
Room 13-5106
Cambridge, MA 02139

Dr. Don Kimberlin
Office of Project Manager
Computerized Training System
Ft Monmouth, NJ

Mr. George Lahey
Navy Personnel Research & Development Center
San Diego, CA 92152

Mr. Hal Peters
Hewlett-Packard
11000 Wolf Road
Cupertino, CA 95014

Dr. Mortenza A. Rahini
Department of Computer Sciences
Michigan State University
East Lansing, MI 48823

Dr. Martin Rockway
Air Force Human Resources Laboratory
Lowry AFB. CO 80230

Dr. Robert Seidel
HumRRO
300 North Washington, Street
Alexandria, VA 22314

Mr. Robert H. Simonsen
System Development Technology
Boeing Computer Services
Seattle, WA 98108

Dr. Wawrence Stolurow
Division of Education Research
State University of New York
Stony Brook, NY 11790

Dr. Paul Tenczar
Computer-Based Educational Research Laboratory
University of Illinois
Urbana, IL 61801

Dr. Karl Zinn
Center for Res4arch in Learning and Teaching
University of Michigan
109 East Madison Street
Ann Arbor, MI 48104

PARTICIPATING ARI STAFF

Mr. James D. Baker
Dr. Beatrice J. Farr
Dr. Frank J. Harris
Dr. Cecil D. Johnson
Dr. Bruce W. Knerr
Ms. Martha Moore
Dr. Leon H. Nawrocki
Dr. Michael H. Strub
Dr. Joseph S. Ward

REFERENCES

Adshead, H. G. Total technology - integrating design with production. In Vliestra, J. & Wielinga, R. F. (eds.) Computer-Aided Design. North Holland, Amsterdam, 1973.

Atkinson, R. C. Ingredients for a theory of instruction. American Psychologist, October 1972, 27, 921-31.

Bunderson, C. V. Instructional software engineering. In Blum, R. (ed.) Proceedings of the Conference on computers in undergraduate science education. College Park, Maryland, Commission on College Physics, University of Maryland, 1971, 379-92.

Bunderson, C. V. & Faust, G. W. Programmed and computer-assisted instrution, in Gage, N. L. (ed.) The Psychology of Teaching Methods 1976, Seventy-fifth Yearbook, Part I, of the National Society for the Study of Education, Chicago, University of Chicago Press, 1976.

Frye, C. H. CAI languages, their capabilities and applications. In Atkinson, R. C. & Wilson, H. A. (eds.) Computer-Assisted Instruction: A Book of Readings, New York, Academic Press, 1969.

Gardner, E. Paper Session, Army Research Institute for the Behavioral and Social Sciences, Authoring Language Conference, Arlington, Virginia, September 1974.

Gero, J. S. A system for computer-aided design in architecture. In Vlietstra, J. & Wielinga, R. F. (eds.) Computer-Aided Design. North Holland, Amsterdam, 1973.

Iverson, K. E. A Programming Language, New York, Wiley, 1962.

Kaplow, R. Description of basic author aids in an organized system for computer-assisted learning. ARI Technical Report (in press)

Kimberlin, D. Paper Session, Army Research Institute for the Behavioral and Social Sciences, Authoring Language Conference, Arlington Virginia, September 1974.

Lahey, G. Paper Session, Army Research Institute for the Behavioral and Social Sciences, Authoring Language Conference, Arlington, Virginia, September 1974.

Merrill, M.D. & Boutwell, R.C. Instructional Development: Methodology and Research. Bringham Young University, Provo, Utah, 1973.

Merrill, M. D. & Wood, N. D. Rules for effective strategies, Orem, Utah: Courseware, Inc., 1976.

Merrill, M. D., Olsen, J. B. & Coldeway, Nancy A. Research support for the instruction strategy diagnostic profile. Orem, Utah: Courseware, Inc. 1976

Peters, H. Roundtable Discussion, Army Research Institute for the Behavioral and Social Sciences, Authoring Language Conference, Arlington, Virginia September 1974.

Powers, G. J. Non-numerical problem solving methods in computer-aided design. In Vlietstra, J. & Wielinga, R. F. (eds.) Computer-Aided Design. North Holland, Amsterdam, 1973.

Rummler, G. A. The economics of lean programming. Improving Human Performance, Fall 1973, 3, 211-16.

Stelzer, J. Roundtable Discussion, Army Research Institute for the Behavioral and Social Sciences, Authoring Language Conference, Arlington, Virginia, September 1974.

Zinn, K. A. Programming conversational use of computer for instruction. In Atkinson, R. C. & Wilson, H. A. (eds.) Computer Assisted Instruction, A Book of Readings, New York, Academic Press, 1969.